IN THE CLAIMS:

Please cancel Claims 75-76 without prejudice, and amend Claims 24, 33, 42, 51, 60, 69 and 74 as follows:

1-23. (cancelled)

24. (currently amended) A method for programming a programmable system comprising a fixed processor portion and a user programmable address arithmetic unit, the method comprising:

writing a first program in a first programming language, said first program configured to implement one or more user-defined address calculation functions in said programmable address arithmetic unit;

compiling said first program;

generating a first executable image, said first executable image adapted for loading into a first ~~program~~ memory coupled to said programmable address arithmetic unit;

writing a second program in a second programming language, said second program configured to implement a desired algorithm, wherein the second programming language ~~consists of~~ comprises a fixed set instructions that make use of a fixed set of addressing modes, wherein at least one of the addressing modes comprises a user defined addressing mode;

compiling said second program into object code, said object code comprising a plurality of machine level instructions drawn from the fixed instruction set of ~~for~~ the processor, said plurality of machine level instructions comprising at least one instruction ~~to control said programmable address arithmetic unit~~ that invokes the user defined addressing mode; and

generating a second executable image, said second executable image adapted for loading into a second program memory coupled to said processor.

25. (previously presented) The method of Claim 24, wherein said address calculation function is invoked by an auto-update addressing mode.

26. (previously presented) The method of Claim 24, wherein said first programming language is a hardware definition language.

27. (previously presented) The method of Claim 26, wherein said second programming language is an assembly language.

28. (previously presented) The method of Claim 26, wherein said second programming language is a high level programming language.

29. (previously presented) The method of Claim 24, wherein said address arithmetic unit provides special purpose circuitry, said method further comprising the step of adapting said first program to use a software library to access said special purpose circuitry.

30. -32. (cancelled)

33. (currently amended) A method of supplying software, comprising:

supplying a first software module comprising instructions drawn from a fixed set of instructions to implement an algorithm using a processor having a fixed portion and a programmable addressing arithmetic unit (PAAU); and

supplying a second software module comprising configuration codes to define the operation of a user-defined addressing mode to be executed in the PAAU; ~~a programmable addressing arithmetic unit (PAAU)~~;

~~whereby~~ wherein at least one instruction in said first module references an operand using said user-defined addressing mode.

34. (previously presented) The method of Claim 33, whereby at least some of said instructions in said first software module are used to program a digital signal processor.

35. (previously presented) The method of Claim 33, whereby said at least one instruction causes an auto-update to be applied to a pointer operand, the operation of the auto-update being defined at least in part by said user-defined addressing mode.

36. (previously presented) The method of Claim 35, further comprising specifying said auto-update using an assembly language mnemonic.

37. (previously presented) The method of Claim 36, whereby said first software module comprises a second instruction, said second instruction being used to specify one of a plurality of user-defined addressing modes to be selected to define, at least in part, the operation of said auto-update.

38. (previously presented) The method of Claim 33, further comprising configuring a programmable logic block within said PAAU using at least a portion of said configuration codes in said second software module.

39. (previously presented) The method of Claim 33, further comprising programming a set of sequential logic operations as implemented by a microsequenced state machine within said PAAU using at least a portion of said configuration codes in said second software module.

40. (previously presented) The method of Claim 33, further comprising programming a crossbar switching element within said PAAU using at least one of said configuration codes in said second software module.

41. (previously presented) The method of Claim 33, whereby said first software module comprises a plurality of subsets of instructions, each subset of instructions to be dispatched to one of a plurality of functional units in a multi-issue processor, the method further comprising dispatching at least one of said subsets to a functional unit comprising said PAAU.

42. (currently amended) A computer-readable medium containing a first software module having a sequence of instruction drawn from a fixed set of instructions to implement an algorithm using a processor having a fixed portion and a programmable addressing arithmetic unit (PAAU), and a second software module containing configuration codes which define the operation of a user-defined addressing mode, said first and second modules implementing the method of:

executing said instructions in said first software module to implement said algorithm; and

using said configuration codes to configure the operation of said user-defined addressing mode to be executed in the PAAU;

~~whereby~~ wherein at least one instruction in said first module references an operand using said user-defined addressing mode.

43. (previously presented) The medium of Claim 42, whereby at least some of said instructions in said first software module are used to program a digital signal processor.

44. (previously presented) The medium of Claim 42, whereby said at least one instruction causes an auto-update to be applied to a pointer operand, and the operation of the auto-update is defined by said user-defined addressing mode.

45. (previously presented) The medium of Claim 44, whereby an assembly language mnemonic is used to specify said auto-update.

46. (previously presented) The medium of Claim 45, whereby said first software module comprises a second instruction, said second instruction specifying one of a plurality of user-defined addressing modes to be selected to define the operation of said auto-update.

47. (previously presented) The medium of Claim 42, whereby at least some of said configuration codes in said second software module are used to configure a programmable logic block within said PAAU.

48. (previously presented) The medium of Claim 42, whereby at least some of said configuration codes in said second software module are used to program a set of sequential logic operations as implemented by a microsequenced state machine within said PAAU.

49. (previously presented) The medium of Claim 42, whereby said configuration codes in said second software module are used to program a crossbar switching element within said PAAU.

50. (previously presented) The medium of Claim 42, whereby said first software module comprises a plurality of subsets of instructions, each subset of instructions to be dispatched to one of a plurality of functional units in a multi-issue processor, whereby one of said subsets is dispatched to a functional unit comprising said PAAU.

51. (currently amended) A method of executing software in a computerized system, said system comprising ~~a processor with~~ a fixed processor portion and a programmable addressing unit (PAAU), a first software module containing a sequence of instructions drawn from a fixed set of instructions to implement an algorithm using a processor having a fixed portion and the PAAU, ~~instructions defining the operation of an algorithm~~ and a second software module containing configuration codes defining the operation of a user-defined addressing mode supplied by said PAAU, the method comprising:

executing instructions in said first software module to implement ~~said~~ an algorithm; and

using said configuration codes to configure the operation of said user-defined addressing mode to be executed in the PAAU;

whereby at least one instruction in said first module references an operand using said user-defined addressing mode.

52. (previously presented) The method of Claim 51, further comprising using at least some of said instructions in said first software module to program a digital signal processor.

53. (previously presented) The method of Claim 51, further comprising:

defining the operation of the auto-update by said user-defined addressing mode; and

causing an auto-update to be applied to a pointer operand using said at least one instruction.

54. (previously presented) The method of Claim 53, further comprising said auto-update using an assembly language mnemonic.

55. (previously presented) The method of Claim 54, whereby said first software module comprises a second instruction, and said method further comprises specifying one of a plurality of user-defined addressing modes to be selected to define the operation of said auto-update using said second instruction.

56. (previously presented) The method of Claim 51, further comprising using at least some of said configuration codes in said second software module to configure a programmable logic block within said PAAU.

57. (previously presented) The method of Claim 51, further comprising using at least some of said configuration codes in said second software module to program a set of sequential logic operations as implemented by a microsequenced state machine within said PAAU.

58. (previously presented) The method of Claim 51, further comprising using said configuration codes in said second software module to program a crossbar switching element within said PAAU.

59. (previously presented) The method of Claim 51, whereby said first software module comprises a plurality of subsets of instructions, each subset of instructions to be dispatched to one of a plurality of functional units in a multi-issue processor, the method further comprising dispatching at least one of said subsets to a functional unit comprising said PAAU.

60. (currently amended) A computerized system adapted for loading a first software module having a plurality of instructions drawn from a fixed instruction set and a second software module having at least one configuration code, the system comprising:

a processor having a fixed portion and a programmable addressing arithmetic unit (PAAU);

~~whereby~~ wherein said processor is adapted to:

(i) execute at least a first one of said plurality of instructions to at least partially implement an algorithm,

(ii) configure a user-defined addressing mode in said PAAU using said at least one configuration code, and

(iii) execute at least a second one of said plurality of instructions, said at least second instruction referencing an operand using said user-defined addressing mode, said second instruction also being executed to at least partially implement said algorithm.

61. (previously presented) The system of Claim 60, whereby at least some of said instructions in said first software module are used to program a digital signal processor.

62. (previously presented) The system of Claim 60, whereby said at least one instruction causes an auto-update to be applied to a pointer operand, and the operation of the auto-update is defined by said user-defined addressing mode.

63. (previously presented) The system of Claim 62, whereby an assembly language mnemonic is used to specify said auto-update.

64. (previously presented) The system of Claim 63, whereby said first software module comprises a second instruction, said second instruction specifying one of a plurality of user-defined addressing modes to be selected to define the operation of said auto-update.

65. (previously presented) The system of Claim 60, whereby at least some of said configuration codes in said second software module are used to configure a programmable logic block within said PAAU.

66. (previously presented) The system of Claim 60, whereby at least some of said configuration codes in said second software module are used to program a set of sequential logic operations as implemented by a microsequenced state machine within said PAAU.

67. (previously presented) The system of Claim 60, whereby said configuration codes in said second software module are used to program a crossbar switching element within said PAAU.

68. (previously presented) The system of Claim 60, whereby said first software module comprises a plurality of subsets of instructions, each subset of instructions to be dispatched to one of a plurality of functional units in a multi-issue processor, whereby one of said subsets is dispatched to a functional unit comprising said PAAU.

69. (currently amended) A computer-implemented method for programming a processor comprising a programmable addressing arithmetic unit (PAAU), the method comprising:

allowing a sequence of instructions defining a program for implementing an algorithm to execute, thereby generating a sequence of addresses;

observing at least a subsequence of said sequence of addresses; and

generating a configuration program for said PAAU, said configuration program defining a user-defined auto-update addressing mode, ~~whereby~~ wherein successive executions of said user-defined auto-update addressing mode is ~~capable of regenerating~~ operative to regenerate said subsequence.

70. (previously presented) The method of Claim 69, whereby said act of observing comprises observing a subsequence that corresponds to an address history sequence of a pointer variable.

71. (previously presented) The method of Claim 70, further comprising defining an auto-update operation using said user-defined addressing mode, said auto-update defining at least one method for advancing from a current address element of said subsequence to a successive address-element of said subsequence.

72. (previously presented) The method of Claim 71, further comprising:

modifying said sequence of instructions by inserting into at least a subset of specified instructions a mnemonic that specifies said auto-update operation, such that when said modified sequence of instructions is executed, said pointer undergoes said observed subsequence of addresses.

73. (previously presented) The method of Claim 72, further comprising executing said sequence of instructions in N cycles and said modified set of instructions in M cycles, with the value of M being less than that of N.

74. (currently amended) A method of executing software in a computerized system, said system comprising a means for processing digital data, said means for processing comprising a fixed processor portion and programmable means for addressing, a first software module containing instructions defining the operation of an algorithm and a second software module containing configuration codes defining the operation of a user-defined addressing mode supplied by said means for addressing, the method comprising the steps of:

executing instructions in ~~a~~ the first software module for implementing said algorithm, wherein the instructions are drawn from a fixed instruction set; and

using said configuration codes ~~for~~ to configure the operation of said user-defined addressing mode;

~~whereby~~ wherein at least one instruction in said first module is used for referencing an operand, said referencing of said operand being accomplished at least in part with said user-defined addressing mode.

75.-76. (cancelled)

Please add new claims 77-79 as follows:

77. (new) For use with a processor having a fixed architecture portion and a programmable address arithmetic unit (PAAU), a method of executing an algorithm by executing a sequence of opcodes assembled based upon an assembly language having a fixed instruction set and at least one mnemonic that refers to an operand in memory according to a user-defined addressing mode, the method comprising:

writing a set of configuration codes into a storage area to define a first operation and a second operation of the user-defined addressing mode;

executing a first instruction that invokes the user-defined addressing mode and calculating a first operand address using the first operation of the user-defined addressing mode;

executing a user-defined addressing mode change instruction that causes the second user-defined addressing mode to be activated into a PAAU; and

executing a second instruction that invokes the user-defined addressing mode and calculating a second operand address using the second operation of the user-defined addressing mode.

78. (new) The method of Claim 77, wherein the user-defined addressing mode change instruction further causes the first user-defined addressing mode to be deactivated.

79. (new) A computerized system comprising:

a processor comprising a fixed architecture portion and a programmable address arithmetic unit (PAAU);

a first software module comprising a collection of opcodes assembled from a fixed assembly language that has a fixed set of instructions, a fixed set of fixed addressing modes, and at least one user-defined auto-update addressing mode, wherein the execution of the first software module results in the implement an application algorithm through a sequence of individual op-code executions defined by a program flow; and

a configuration data module comprising at least one configuration code adapted to configure the operation of at least one user-defined auto-update addressing mode in the PAAU;

wherein at least a subsequence of the sequence of individual op-code executions makes reference to an operand using the user-defined auto-update addressing mode, and the execution of successive instructions in the subsequence of instructions results in a addressing pattern to be generated, wherein the addressing pattern is nonlinear and is dependent on the application algorithm, and wherein the user-defined auto-update addressing mode is configured to generate the addressing pattern using less instruction cycles than would be possible by using a combination of instructions involving the fixed set of addressing modes.